

## **A SYSTEM AND METHOD OF DESIGNING, TESTING, AND EMPLOYING GRAPHICAL COMPUTER CODE**

### **Field of the Invention**

The present invention relates generally to the field of software for developing user interfaces and more particularly to a system and method for designing, testing, and employing graphical computer code.

### **Background of the Invention**

Computer software and systems for creating user interface prototypes are currently in existence. The existing software allows the user to create new user interfaces, limited only by the user's imagination and the software's graphical limitations. The prototype interfaces created with this software save time and money by allowing the creation of control panels for hardware and software with a computer, instead of requiring the user to manufacture time and labor intensive prototype hardware. The software programs allow designers to create user

interfaces without knowledge of complicated programming languages. Once the interface is created, the computer code necessary to drive the actual interface needs to be tailored to the operating parameters of the actual device. However, the computer code that would drive these user interfaces has to be created separately, effectively reinventing the user interface already created with the software. This duplicates effort and wastes resources. One solution has been to create graphical code templates, written in a high-level computer language, that are copied into an overall graphics program. Unfortunately, there are a limited number of templates and the ability to tailor the templates is extremely limited.

Thus there is a need for a system that allows not only the creation of new user interfaces that meet the requirements of new products, but also automatically creates the computer code to drive the user interface in the new product.

### **Summary of the Invention**

A system for designing, testing, and employing graphical computer code includes a graphics editor for creating a graphical display. The graphical display is made up of a plurality of graphical objects constructed by the graphics editor. A

translator creates a high-level computer language code of the graphical object.

The translator is connected to the graphics editor. A compiler receives the high level computer language code from the translator.

### **Brief Description of the Drawings**

FIG. 1 is a block diagram of a system for designing, testing, and employing graphical computer code in accordance with one embodiment of the invention;

FIG. 2 is a block diagram of a system for designing, testing, and employing graphical computer code in accordance with one embodiment of the invention;

FIG. 3 is a block diagram of a system for designing, testing, and employing graphical computer code in accordance with one embodiment of the invention;

FIG. 4 is a flow chart of a method of designing, testing and employing graphical computer code in accordance with one embodiment of the invention; and

FIG. 5 is a block diagram of a translation system for designing, testing, and employing graphical computer code in accordance with one embodiment of the invention.

### **Detailed Description of the Drawings**

A system for designing, testing, and employing graphical computer code includes a graphics editor for creating a graphical display. The graphical display is made up of a plurality of graphical objects constructed by the graphics editor. A translator creates a high-level computer language code of the graphical object. The translator is connected to the graphics editor. A compiler receives the high-level computer language code from the translator. An advantage of this systemic that it automatically creates code that is ready to be installed in the finished product.

FIG. 1 is a block diagram of a system 10 of designing, testing, and employing graphical computer code in accordance with one embodiment of the invention. The graphics editor 12 creates a graphical display made up of a plurality of graphical objects constructed by the graphics editor. A translator 14 is connected to the graphics editor 12. The translator 14 creates a high-level computer language code from the graphical objects. A compiler 16 receives the high-level computer language code from the translator 14. In one embodiment, a run time system is connected to the graphics editor 12 and is designed to execute a graphical design. In another embodiment, a control editor is connected to the

graphics editor 12 and allows a user to embed logic into the graphics. Design elements may be assembled from primitive graphical objects. These design elements may be assigned particular behaviors in response to assigned stimuli, or the design elements may be created to supply stimuli to direct a particular display behavior.

In another embodiment, a library of graphical is connected to the graphics editor 12. These graphical objects are created using the graphics editor 12 and are stored in the library. The translator 14 has a number of options, such as translating a graphical objects input stimulus, translating a graphical objects control logic, allowing dynamic memory allocation, and selecting a subset of graphical objects. The translator 14 examines the graphical object selected to be translated. The data structure's primitive graphical objects, from which the generated code is built, are then sized based upon the number and type of graphical objects selected. This allows the system to create a final, compiled version of the code in as small a space as is possible, or any reasonable size limitation.

FIG. 2 is a block diagram of a system 20 for designing, testing, and employing graphical computer code in accordance with one embodiment of the invention. A graphics environment 22 creates a graphical display made up of a plurality of graphical objects constructed by the graphics environment 22. A

translator 24 is connected to the graphics environment 22 and creates a high-level computer language code. A control editor 26 is connected to the graphics environment 22, and allows a user to embed logic into the graphics. In one embodiment, the graphics environment 22 contains a library of components, e.g., push-buttons, knobs, sliders, etc., and a run time system, and the run time system is designed to execute a graphical design, e.g., the run time system may operate a real time stimulation of the user interface, as described by the assembly of the components from the library, and as directed to operate by the stimulus and response programming. Such a system will allow a user to see exactly what the final interface will look like. More importantly, the system will allow the user to quickly and cheaply test the function of each feature of the user interface, even allowing the connection of the graphical interface to actual, and not simulated, hardware to fully test the interface's functionality.

FIG 3 is a block diagram of a system 30 for designing, testing, and employing graphical computer code in accordance with one embodiment of the invention. The graphics editor 32 creates a graphical display made up of a plurality of graphical objects 34, e.g., push-buttons, knobs, sliders, etc. and previously-constructed display components, constructed by the graphics editor 32. A translator 36 is connected to the graphics editor 32 and creates a high-level

computer language program, in a language such as C, from the graphics editor graphical objects. The translator has a number of options such as selecting a graphical objects input stimulus 38 for translation, translating a graphical objects control logic 40, allowing dynamic memory allocation 42, and sizing a data structure 44. A compiler 46 receives the high-level computer language code from the translator 36. A run time system 48 executes a graphical design. A control editor 50 is connected to the graphics editor 32. A library of graphical objects 52 is connected to the graphics editor 32.

From a user's perspective the system is remarkably easy to use. The user assembles a graphical display from shapes and components in the library 52. An input/stimulus is defined and a behavior/response is defined for that input. This stimulus and response correlation is repeated throughout the display range of the graphical display. These behavior states are stored in the system's memory. The system may be directed to interpolate a behavior response when the stimulus input is between programmed values. When the stimulus/response definition is complete, the tools required for that phase of graphical display construction may be closed or hidden from view on the computer screen, leaving just the run time system in view. The run time system 48 allows the user to operate the interface within the limits of its programming to test the functionality of the program. The



system may operate a simulated hardware unit or may be connected to a real hardware unit to insure the program's compatibility. If the interface operates as desired, the system may be directed to send the high-level computer language code to the compiler 46, to create a compiled code for the selected hardware. If the interface does not operate as desired, or if the design is found to be awkward to use, it is a simple matter to rearrange, remove, or replace the undesirable design elements, using the same software tools that were used to create the graphical display.

FIG. 4 is a flow chart of a method of designing, testing, and employing graphical computer code in accordance with one embodiment of the invention. The process starts, step 60, by creating a graphical display with a graphics editor at step 62. The system translates a graphical code of the graphical display into a high level computer language code at step 64. At step 66 the system compiles the high level computer language code which ends the process at step 68. In one embodiment, the system identifies a target processor for a compiler. In another embodiment, the system examines a plurality of objects to be translated, and determines if dynamic memory allocation is selected. When dynamic memory allocation is not selected, the system allows the user to select memory allocation sizes. In another embodiment, the system translates a graphical objects input

stimulus, translates a graphical objects control logic, and translates a graphical objects graphical representation. In another embodiment, the system creates an animation sequence by example, and creates an animation input stimulus.

FIG.5 is a block diagram of a translation system 70 for designing, testing, and employing graphical computer code in accordance with one embodiment of the invention. An array builder 72 constructs a data array 74 from a plurality of graphical objects 76, including pushbuttons 78, knobs 80, and sliders 82. A code builder 84 translates a high-level computer language code from the data array 74 a library of computer code operations 86 is connected to the code builder 84. In one embodiment, the code builder 84 includes a data sizing function 88. In another embodiment, the library of files for graphics drawing and files for generating animation 90, stimulus 92, and control 94 codes. The code builder 84 includes a dynamic memory allocation choice 96. The code builder 84 includes a dynamic memory allocation choice 96. The code builder 84 assembles a high-level computer language program 98.

The methods described herein can be implemented as computer-readable instructions stored on a computer-readable storage medium that when executed by a computer will perform the methods described herein.

While the invention has been described in conjunction with specific

embodiments thereof, it is evident that many alterations, modifications, and variations will be apparent to those skilled in the art in the light of the foregoing description. Accordingly, it is intended to embrace all such alterations, modifications, and variations in the appended claims.